UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/059,427	01/29/2002	Jeroen Anton Johan Leijten	260695	6839

23460 7590 03/19/2009
LEYDIG VOIT & MAYER, LTD
TWO PRUDENTIAL PLAZA, SUITE 4900
180 NORTH STETSON AVENUE
CHICAGO, IL 60601-6731

EXAMINER	
PETRANEK, JACOB ANDREW	
ART UNIT	PAPER NUMBER
2183	
MAIL DATE	DELIVERY MODE
03/19/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte JEROEN ANTON JOHAN LEIJTEN

Appeal 2008-2554
Application 10/059,427
Technology Center 2100

Decided:[1] March 19, 2009

Before JOHN C. MARTIN, ALLEN R. MACDONALD, and
JEAN R. HOMERE, *Administrative Patent Judges.*

MARTIN, *Administrative Patent Judge.*

DECISION ON APPEAL

[1] The two-month time period for filing an appeal or commencing a civil action, as recited in 37 C.F.R. § 1.304, begins to run from the decided date shown on this page of the decision. The time period does not run from the Mail Date (paper delivery) or Notification Date (electronic delivery).

STATEMENT OF THE CASE

This is an appeal under 35 U.S.C. § 134(a) from the Examiner's rejection of claims 1-9, which are all of the pending claims.

We have jurisdiction under 35 U.S.C. § 6(b). We affirm.

A. *Appellant's invention*

Appellant's invention relates to instruction fetching in a computer (Specification 1:2) and more particularly to the fetching of VLIW (Very Long Instruction Word) instructions that have been compressed using variable-length instruction formats. *Id.* at 1:5-6.

In accordance with Appellant's invention, each stored compressed instruction word has a header containing at least one fetch control bit. *Id.* at 5:11-11-13.

Appellant's Figure 2 as filed is reproduced below.[2]

[2] The misspelling of "prefetch" is corrected by a proposed drawing amendment filed with the Brief.

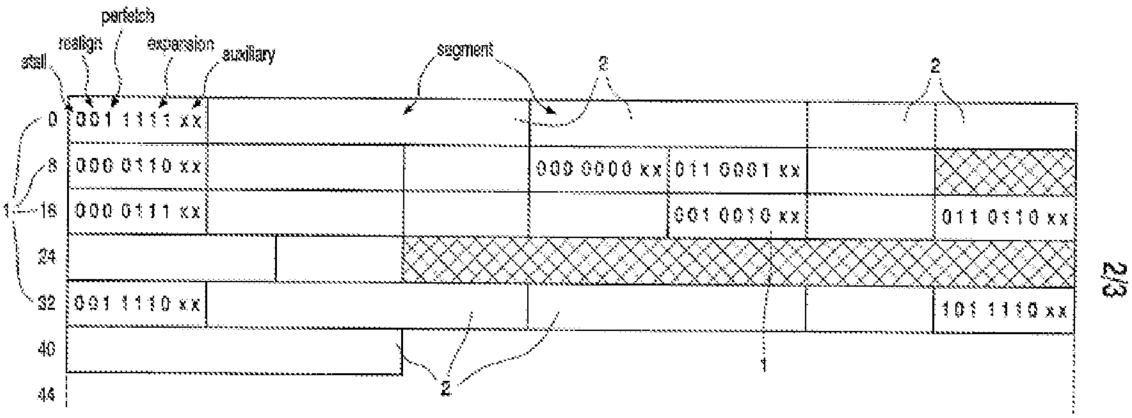

FIG. 2

Figure 2 shows an example of how VL^{2}IW instructions are stored in program memory. *Id.* at 11:25. The first three bits of the header 1 of each instruction word are the stall bit, realign bit, and prefetch bit, respectively. The stall bit indicates whether the present instruction is spread over more than one line in case the last of those lines has not yet been fetched. *Id.* at 8:28-30. The realign bit indicates whether the instruction is followed by padding, which should not be executed. *Id.* at 8:23-25. The prefetch bit indicates whether the next instruction to be executed is contained in the memory line presently received from memory 30, or whether part or whole of the instruction is contained in a next memory line. *Id.* at 8:16-18.

The values of the stall bit, realign bit, and prefetch bit are determined during compilation of the variable length instructions, taking into account where the instructions will be stored in the instruction memory. See *id.* at 14:24-26 and Fig. 4. For example, in step 44 of the flowchart depicted in

Figure 4, it is determined whether the length of the current instruction is such that when added to the position of the end of the previous instruction, the current instruction will transgress the end of a memory line. *Id.* at 15:3-5.

B. The claims

The independent claims before us are claims 1 and 8, of which claim 1 reads:

> 1. A computer system with a processing unit and a memory, the processing unit being arranged to fetch memory lines from the memory and execute instructions from the memory lines, each memory line being fetched as a whole and being capable of holding more than one instruction, at least one instruction comprising information, inserted at compile time, that signals explicitly how the processing unit, when processing the instruction from a current memory line, should control how a part of processing is affected by crossing of a boundary to a subsequent memory line, the processing unit being arranged to respond to the information by controlling said part as signaled by the information.

Claims App., Br. 15.

Claim 8 is a method claim that tracks "system" claim 1.

C. The references and rejection

The Examiner relies on the following references:

Miller et al. (Miller)	US 5,819,058	Oct. 6, 1998
Keller et al. (Keller)	US 6,546,478 B1	Apr. 8, 2003

Mohamed et al. (Mohamed)	US 6,684,319 B1	Jan. 27, 2004

Claims 1, 4, and 6-8 stand rejected under 35 U.S.C. § 102(b) for anticipation by Miller.

Claims 2, 3, and 9 stand rejected under § 103(a) for obviousness over Miller in view of Mohamed.

Claim 5 stands rejected under § 103(a) for obviousness over Miller in view of Keller.

Appellant separately argues the merits of only claims 1 and 8.[3]

THE ISSUES

Appellant has the burden to show reversible error by the Examiner in maintaining the rejection. *See In re Kahn*, 441 F.3d 977, 985-86 (Fed. Cir. 2006) ("On appeal to the Board, an applicant can overcome a rejection by showing insufficient evidence of *prima facie* obviousness or by rebutting the *prima facie* case with evidence of secondary indicia of nonobviousness.") (quoting *In re Rouffet*, 149 F.3d 1350, 1355 (Fed. Cir. 1998)).

Appellant argues that Miller fails to satisfy two limitations that appear in each of claims 1 and 8. Specifically, Appellant argues that the Examiner erred in finding that the EP/NEP field relied on by the Examiner

[3] Although claim 4 is not separately argued, the copy of that claim in the Claims Appendix to the Brief is incorrect, as noted by the Examiner. Answer 2. A correct copy of that claim appears in the Claims Appendix to the Reply Brief.

(1) corresponds to the recited "information" and (2) was "inserted at compile time."

PRINCIPLES OF LAW

"To anticipate a claim, a prior art reference must disclose every limitation of the claimed invention, either explicitly or inherently." *In re Schreiber*, 128 F.3d 1473, 1477 (Fed. Cir. 1997).

FINDINGS OF FACT REGARDING MILLER

Miller discloses a system and method for compressing and decompressing very long instruction words that are stored in a memory within a processor. Miller, col. 1, ll. 6-10.

Figure 1 of Miller shows a very long instruction word processor that includes the following processing units: an execution control unit (ECU) 26, a multiplier unit (MUL) 30, an arithmetic logic unit (ALU) 32, a register control unit (RCU) 34, and a memory unit (MEM) 36. *Id.* at 3, ll. 5-60.

Miller's Figure 2 is reproduced below.

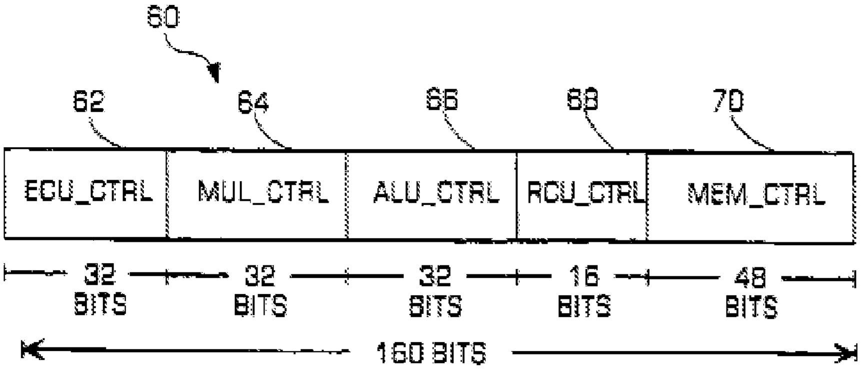

FIGURE 2

Figure 2 is an example of a VLIW 60 that includes a plurality of instruction words of various lengths that individually control respective ones of the processing units. *Id.*, col. 4, ll. 38-57. As shown in Figure 4, discussed in more detail below, each instruction word that comprises the VLIW may be separately compressed into a compressed instruction word.[4]

Figure 3A is reproduced below.

[4] Every VLIW need not contain an active instruction word for every processing unit (*id.*, col. 6, ll. 7-10). The inactive instruction words in a VLIW may be represented as default "NO_OP" (NO-OPERATION) instructions (*id.*, col. 6, ll. 10-11). Figure 4 shows a VLIW 120 containing three NO_OP instructions.

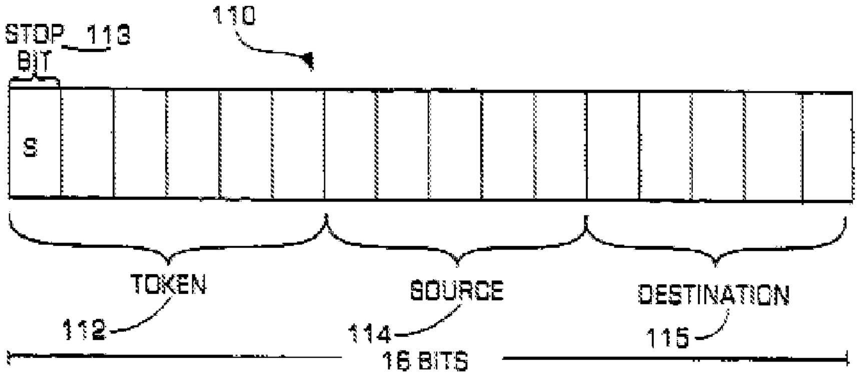

FIGURE 3A

Figure 3A is a diagram of the format of a 16-bit compressed instruction word 110. *Id.*, col. 5, ll. 8-11. "The stop bit is set to '1' if this particular instruction is the last instruction within a compressed instruction packet." *Id.*, col. 5, ll. 30-32. Otherwise, presumably, the stop bit is "0."

Figure 3B is reproduced below.

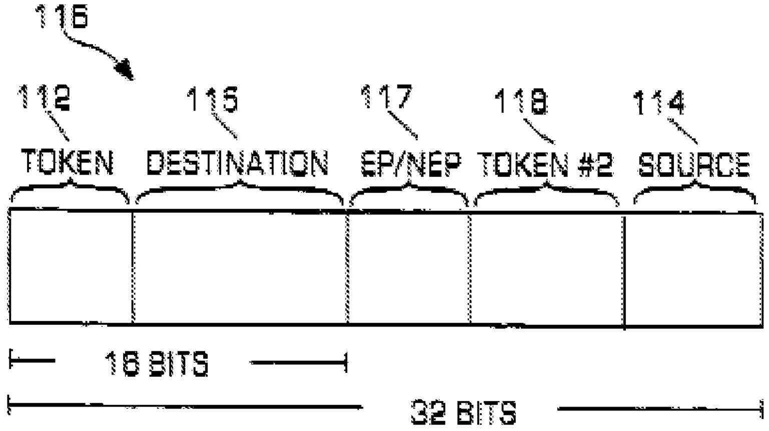

FIGURE 3B

Figure 3B is an example of a format of a 32-bit compressed instruction 116. *Id.*, col. 6, ll. 20-21. For any compressed instruction longer than 16 bits, the location of the end of packet indicators that indicate whether the compressed instruction is the last compressed instruction within a packet has been moved. *Id.*, col. 6, ll. 25-28. As shown, there may be a end-of-packet (EP)/not-end-of-packet (NEP) field 117 and a second token field 118 in the longer compressed instruction. *Id.*, col. 6, ll. 28-31. The EP/NEP field permits the system to determine whether the particular compressed instruction is at the end of a compressed instruction packet and performs the same function as the stop bit 113 in the 16-bit long compressed instruction. *Id.*, col. 6, ll. 31-35.

Figure 7 is a diagram of the instruction memory 172 storing a plurality of compressed instruction packets. *Id.*, col. 10, ll. 36-37. The memory consists of first (left) and second (right) 64-bit wide memories 174[5] and 176 (*id.*, col. 10, ll. 38-40), which are separately addressed using addresses ALEFT and ARIGHT. *Id.*, col. 10, ll. 3-8. In operation, 128 bits are read out of the instruction memory during each clock cycle. *Id.*, col. 10, ll. 43-44.

As shown in Figure 7, instruction packets IP1 and IP2 are small enough to fit in the first memory line of the left and right memories along with IP0. Packet IP3, on the other hand, begins in the first line of the right memory and ends in the second line of the left memory. As shown in Figure 8, IP3 includes memory addresses 6 and 7 in the first line (ARIGHT = 0) of the right memory and memory addresses 8 and 9 in the second line

[5] Mislabeled "171" in Figure 7.

(ALEFT = 1) of the left memory.

WHETHER THE EP/NEP FIELD RELIED ON BY THE EXAMINER CORRESPONDS TO THE RECITED "INFORMATION"

In the Final Action (at 4-5), the Examiner read the recited "information . . . that signals explicitly how the processing unit, when processing the instruction from a current memory line, should control how a part of processing is affected by crossing of a boundary to a subsequent memory line" on EP/NEP field 117, Figures 7 and 8, and column 11, line 58 to column 12, line 9, which lines explain, *inter alia*, that

> [t]he addressing system . . . determines that IP3 crosses the 128-bit boundary, so the Aleft address is incremented by one so that addresses 8, 9, 10, and 11 may be accessed. Thus, IP3 is read out of locations 6, 7, 8, and 9. To read out IP3, the data in the second memory is read out first and then the data in the first memory is read out.

Miller, col. 12, ll. 4-9.

Appellant responded to this position of the Examiner by characterizing the cited portions of Miller as "merely concerned with an addressing system using the proper address" (Br. 10) without explaining why the recited "information" does not read on the EP/NEP field of instruction packet IP3.

In the Answer, the Examiner provided a more detailed explanation of the significance of EP/NEP field 117:

> Looking at figure 8, IP3 is located in memory segments 6-9 in memory lines 0 and 1. It's assumed that IP3 contains two 32-bit compressed instructions for this example. The 32-bit compressed instruction in memory segments 6-7 contains element 117 signaling not-end-of-packet since another compressed 32-bit instruction is in memory segments 8-9. Thus, element 117 will indicate that the subsequent memory line must be read out of the memory to get the whole VLIW packet to completely decompress the VLIW packet.)[.]

Answer 4.

Appellant responded by arguing that

> element 117 of Miller, namely an end-of-packet (EP), not-end-of-packet (NEP), or a stop bit, is not equivalent to information that "signals explicitly how the processing unit, when processing the instruction from a current memory line, should control how a part of processing is affected by crossing of a boundary to a subsequent memory line," as required by Independent Claims 1 and 8.

Reply Br. 2. However, Appellant provided no reasoning in support of this conclusory argument and thus has failed to show that the Examiner erred in reading the recited "information" on EP/NEP field 117 in the above manner.

WHETHER THE NP/NEP PACKET RELIED ON BY THE EXAMINER WAS "INSERTED AT COMPILE TIME"

In the Final Action, the Examiner read the "inserted at compile time" claim language on column 6, lines 53-58, which explains: "FIG. 4 is a diagram of an uncompressed very long instruction word (VLIW) 120, and a corresponding compressed instruction packet 121 in accordance with the invention. The very long instruction word 120[] may be compressed, in accordance with the invention, by a compiler or an assembler." Final Action 5. Appellant responded in the Brief by arguing that "[i]nformation, inserted at compile time, that signals explicitly how the processing unit, when processing the instruction from a current memory line, should control how a part of processing is affected by crossing of a boundary are nowhere taught or suggested in Miller." *Id.* at 12. In the Answer, the Examiner further explained that

> Figure 4 shows how an uncompressed very long instruction word (VLIW) can be compressed down, which can be done by a compiler (Miller: Column 6 lines 56-58). Since the compression is performed by the compiler, then the end-of-packet (EP)/not-end-of-packet (NEP) field (element 117) of figure 3b must be inserted by the compiler. Thus, the compiler inserts information into at least one instruction.

Answer 10. Appellant responded by noting that the EP/NEP field performs the same function as the stop bit does in a 16-bit compressed instruction word (Reply Br. 2-3) and arguing that

> [a]s clearly shown in FIG 4, the "stop bit 128 and the control bits 130 of the uncompressed instruction are compressed into the 6 bit stop bit and token field 132, 134." (Column 7, lines 22- 23, emphasis added) That is, the stop bit is already present in the uncompressed instruction and is merely compressed. Miller simply does not teach or suggest that the stop bit or an EP/NEP is inserted at compile time.

Id. at 3.

We are not persuaded by this argument. Although uncompressed instruction words 122 and 126 in VLIW instruction word 120 contain stop bit positions 128 and 146, respectively, Miller indicates that the stop bit values are *set* during the compression process, which converts the uncompressed instruction words 122 and 126 into compressed instruction words 127 and 144, respectively. *See* Miller, col. 5, ll. 31-33 ("The stop bit is set to '1' if this particular instruction is the last instruction within a compressed instruction packet."). Thus, stop bits 132 and 154 in the compressed instruction words 127 and 144 are "set" to 0 and 1, respectively, during compiling for compression.

Furthermore, assuming for the sake of argument that Appellant is correct to assert that the stop bit values are present in uncompressed instruction words 122 and 126 prior to the compression operation, the "at the time of compiling" claim language is broad enough to read on the process of initially compiling VLIW instruction word 120 to include those uncompressed instruction words.

CONCLUSION

Appellant has failed to show the Examiner erred in finding that the EP/NEP field relied on by the Examiner (1) corresponds to the recited "information" and (2) was "inserted at compile time."

DECISION

The rejection of claims 1 and 8 under 35 U.S.C. § 102(b) for anticipation by Miller is affirmed, as is the rejection on that ground of unargued dependent claims 4, 6, and 7. 37 C.F.R. § 41.37(c)(1)(vii) (2006).

Because the rejection of claims 2, 3, and 9 under § 103(a) for obviousness over Miller in view of Mohamed and the rejection of claim 5 under § 103(a) for obviousness over Miller in view of Keller are not separately argued, the rejections of those claims are also affirmed. *In re Nielson*, 816 F.2d 1567, 1572 (Fed. Cir. 1987).

No time period for taking any subsequent action in connection with this appeal may be extended under 37 C.F.R. § 1.136(a). *See* 37 C.F.R. §§ 41.50(f) and 41.52(b).

AFFIRMED

rwk

LEYDIG VOIT & MAYER, LTD
TWO PRUDENTIAL PLAZA, SUITE 4900
180 NORTH STETSON AVENUE
CHICAGO IL 60601-6731